This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks

Donghui Hu, Liang Wang, Wenjie Jiang, Shuli Zheng, Bin Li

***Abstract*—The security of image steganography is an important basis for evaluating steganography algorithms. Steganography has recently made great progress in the long-term confrontation with steganalysis. To improve the security of image steganography, steganography must have the ability to resist detection by steganalysis algorithms. Traditional embedding-based steganography embeds the secret information into the content of an image, which unavoidably leaves a trace of the modification that can be detected by increasingly advanced machine-learning-based steganalysis algorithms. The concept of steganography without embedding (SWE), which does not need to modify the data of the carrier image, appeared to overcome the detection of machine-learning-based steganalysis algorithms. In this paper, we propose a novel image SWE method based on deep convolutional generative adversarial networks (DCGANs). We map the secret information into a noise vector and use the trained generator neural network model to generate the carrier image based on the noise vector. No modification or embedding operations are required during the process of image generation, and the information contained in the image can be extracted successfully by another neural network, called the extractor, after training. The experimental results show that this method has the advantages of highly accurate information extraction and a strong ability to resist detection by state-of-the-art image steganalysis algorithms.**

***Index Terms*—steganography, without embedding, coverless, generative adversarial networks**

I. Introduction

IMAGE steganography is the art and science of hiding secret information in a carrier image so that a receiver can recover the secret information while a warder cannot detect the presence of the secret information. Currently, most image steganography methods achieve the goals of steganography by embedding secret information into carrier images, which unavoidably leaves evidence of distortion. Steganalysis algorithms are developed to determine whether an image has been embedded based on statistical features of distortion [1]. Adaptive steganography algorithms are proposed to minimize the embedding distortion by using syndrome-trellis codes [2–4]; however, they can also be detected by more advanced machine-learning-based steganalysis algorithms, such as rich models [5–7] and deep-learning-based methods [8–10]. To further reduce the risk of being detected by steganalyzer, scholars have proposed the novel concept of steganography without embedding (SWE) [11].

Donghui Hu, Liang Wang, Shuli Zheng and Wenjie Jiang are with the School of Computer and Information, Hefei University of Technology, Hefei 230009, China (e-mail: hudh@hfut.edu.cn; wang_liangx@163.com; zsl251@163.com; 254751910@qq.com).

B. Li is with the College of Information Engineering, Shenzhen University, Shenzhen 518060, China (email: ibin@szu.edu.cn).

In SWE, the information of the cover image itself is used to represent the secret information. Digital images already contain rich information, such as pixel brightness, color, texture, edge, contour and high-level semantics, and pixels themselves. With appropriate description and processing of this information, it is possible to create a mapping relationship between this information and the secret information to be hidden. We simply need to obtain suitable digital images that can carry information via mapping and designing a method for this type of mapping. Several SWE methods have recently been proposed [12–16]. In some of these works, SWE methods were designed by mapping the secret information from the semantic features of natural images with the help of bag of words [12] or image hashing [13] [14]. This type of method has the weakness of limited steganography capacity and the need to construct a large natural image database. In other works, SWE methods were proposed to map the secret information from a class of texture synthesis [15, 16] with the help of carefully designed reversible mathematical functions. The steganography capacity of the second type of SWE method is also limited. Another drawback of this type of methods is that the carrier images are a fixed type of texture synthesis image or fingerprint image, which may alert the warder in real-world steganography applications. In this paper, we utilize the newly proposed GANs to design new type of SWE to increase the steganography capacity while providing various “natural” carrier images.

GANs, proposed by Goodfellow in [17], consist of a generative model and a discriminative model. The henerative model deceives the discriminative model via generated images that appear like real images while the discriminative model judges whether the images are real or unreal. The ability of the two models is strengthened via adversarial iterative training. The studies in [18] and [19] improved GANs to produce images that look more natural, and many optimized models have been derived from GANs. The Google Brain team compared the performance of GANs with that of other superior optimized models. The results revealed in their report [20] indicated that the original GANs model showed the best performance. Deep convolutional generative adversarial networks (DCGANs) are an extension of GANs in which the models are deep convolutional networks [21]. Currently, GANs are widely used for image generation [22] [23] and image restoration [24] but seldomly used in image steganography, especially SWE. The research in [25–27] proposed different methods using GANs for embedding based image steganography to minimize the embedding distortion and to enhance the ability to resist steganalysis.

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

In this work, we propose a new SWE method based on DCGANs. We establish a relationship between the secret information and a noise vector, which is the input of DCGANs. Stego images are generated by the generator in DCGANs according to preprocessed secret information, and no information is embedded in stego images during the generation period. Another convolutional neural networks (CNNs) called the extractor is designed to recover the secret information from these stego images.

The main contributions of this paper includes:

- We propose a new approach of using GANs for steganography, it is different from previous approaches. Previous steganography methods using GANs are embedding based ones, where traditional embedding skills such as LSB-match are used in the to hide the secret information. While in our method, we use DCGANs to a generate cover image only according to secret information, and the cover image (also is the stego image) itself already "contained" secret information, there is no embedding process in our method.
- Highly accurate secret data recovery is achieved by the carefully designed training of the DCGANs in different phases and by mapping the secret information into noise vectors. In the first phase of training, the mapped noise vectors are trained into "real" images (stego images); in the second phase, we train the DCGANs to "correctly" extract the noise vectors from the stego images.
- Compared with other non-embedding-based image steganography methods, the proposed method significantly improved the steganography performance.
- Our method can resist detection by some state-of-the-art steganalysis and image forensics algorithms when the training image set is kept as a secret.

The rest of this paper is organized as follows: Section II presents the related work on image steganography and steganalysis algorithms. Section III describes the proposed method. Section IV presents the experimental results and analysis. Conclusions are presented in Section V.

II. Relevant Work

A. Embedding-Based Steganography

Most traditional image steganography methods are embedding based, in which the secret information is embedded into the carrier image via a specific type of modification. How to improve the embedding capacity while minimizing embedding distortion is the main driving force of the research in this field. Mielikainen et al. proposed an improved version of least-significant-bit (LSB) matching that enables embedding the same payload as LSB matching but with fewer changes to the cover image [28]. Pevny et al. introduced highly undetectable stego (HUGO) in [2], a new embedding algorithm for spatial domain digital images. The main design principle is to minimize a distortion function that is defined according to the weighted sum of differences between the feature vectors extracted from a cover image and its stego version in subtractive pixel adjacency matrix (SPAM) [29] feature space. Wavelet obtained weights (WOW), presented by Holub et al. in [3], embeds the payload in cover images while following the rule that the more complex the texture of a region of an image is, the more pixel values within that region that will be modified. S-UNIWARD [4], proposed by Holub et al., works in the spatial domain to achieve a similar goal: to embed more information in noisy or complex texture regions of the cover image. The source of the cover image is important to take into account the security of image steganography, but the greatest threat to embedding-based steganography is steganalysis, a method for detecting stego images from cover images. The ability of steganalysis has increased sharply with the development of statistical analysis and machine learning.

B. Development of Steganalysis

Steganalysis algorithms are designed to determine whether a given image is a stego image. Fridrich et al. presented a reliable and accurate method for detecting LSB non-sequential embedding in digital images [30]. Shi et al. proposed a steganalysis scheme for JPEG steganography [31], where a Markov process is applied to formulate the JPEG steganalysis features. Pevny et al. proposed a new merged feature set with Markov and discrete cosine transform features for JPEG image steganalysis [32]. Pevny et al. presented a SPAM to compute the features for steganalysis of stego images in the spacial domain [29], where the first-order and second-order Markov chains are used to model the differences between adjacent pixels. Fridrich et al. proposed a general methodology for the steganalysis of digital images based on the concept of a rich model consisting of a large number of diverse submodels [5]. Goljan et al. proposed an extension of the spatial rich model for the steganalysis of color images [7]. With the recent rapid development of deep learning, steganalysis based on deep learning has been proposed. Qian et al. proposed a CNNs model for steganalysis that can automatically learn feature representations in CNNs layers and capture complex dependencies, which are useful for steganalysis [10]. Zeng et al. [9] proposed a hybrid deep-learning framework for JPEG image steganalysis by incorporating the domain knowledge of rich models. Hu et al. proposed a selection region and a combined CNNs-based method for adaptive steganalysis [33]. These machine-learning-based steganalysis methods are becoming increasingly powerful, which poses a major challenge to embedding-based steganography.

C. Image Steganography without Embedding

SWE, a new type of steganography focused on how to establish the relationship between secret information and cover images was recently proposed to provide a new way to avoid the detection of machine-learning-based steganalysis. Two methods are currently used to achieve SWE: the cover-selection-based approach and the cover-synthesis-based approach [34]. The cover-selection approach constructs an image library collecting a set of natural images and then establishes the relationship between the secret information and the images in the library. Each message can be mapped by a single image or a set of images. The cover-synthesis approach first generates a new cover image driven by the secret information. Zhou

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

et al. [13] proposed an SWE framework by constructing an image database composed of a large number of images indexed according to their hash sequence generated by a robust hashing algorithm. Then, the binary secret data are divided into a number of segments. For each segment, the method selects the image from the database with the hash value equal to the value of the segment. Zhou et al. proposed another SWE method by using the bag-of-words (BOW) model [12]. In this method, visual words are extracted from an image set by the BOW model; then, the mapping between text information keywords and the visual words of images is established. According to the mapping relation, a set of sub-images with visual words related to the text information is found. The images containing these sub-images are used as stego images for secret communication. In [14], a robust image hash was proposed to create the relationship between the secret data and the image. The secret data are divided into segments to match the images in the local image library according to the image hash values. Due to the designed robust image hashing, the steganography capacity is improved relative to that in previous work, and the carrier image is resistant to common image attacks. The drawbacks of the cover-selection-based approach are obvious: the steganography capacity is limited and a large local image library is required. Methods to generate texture images based on secret information were proposed in [15, 16, 35] from the perspective of cover-synthesis. Hirofumi et al. presented an approach [35] to image coding that first paints a regularly arranged dotted pattern, and then camouflages the dotted pattern using the same texture sample while preserving quality comparable to that of existing synthesis techniques. Xu et al. presented the stego-texture [15], a unique texture synthesis method to transform an input image or text message into an intricate texture image. The stego-texture is generated by a real-time texture synthesis system from a given message via a reversible mathematical function, and the hidden message can be retrieved by a decrypter. Wu et al. proposed another steganography approach by using reversible texture synthesis [16] where smaller texture images are re-sampled to create a new texture image with a similar local appearance and an arbitrary size. The texture synthesis process is weaved into steganography to conceal secret messages. However, state-of-the-art cover-synthesis-based approaches have the same weakness: the synthetic images used for these SWE schemes are a special class of images (such as a class of texture images), and it is unusual to send a large number of this type of image, which may alert a warden.

D. GANs for Image Steganography

The emergence of GANs has provided new approaches to achieve image steganography. Hayes et al. proposed an image steganography method using GANs [25] that is competitive with state-of-the-art steganography techniques, as well as a robust steganalyzer that performs the discriminative task of determining whether an image contains secret information. Suppose that Alice, Bob and Eve are three neural networks. Alice is trained to generate stego images, Bob can extract secret data from stego images, and Eve is a steganalysis tool. Information is embedded into the least-cost location in each training period. However, the experimental results show that the trained steganalysis tool Eve does not achieve the distinguishing goal with a probability of 0.5. Therefore, stego images produced by this type of steganography method are easily detected by other steganalysis algorithms, and the secret data cannot be decoded completely by Bob.

Volkhonskiy et al. presented a new adversarial training structure for steganography called SGANs [26], which accounts for not only the authenticity of the generated images but also the resistance to the detection. The proposed model trains a cover image container that can generate secure steganography cover images to deceive a given type of steganalysis. The LSB-match embedding scheme is applied to SGANs to produce the corresponding steganography container. However, different embedding schemes may cause SGAN retraining. The resistance to detection of current steganalysis algorithms is not reported in this paper.

Tang et al. proposed an automatic stego distortion learning framework by using GANs consisting of two adversarial subnetworks [27]. The proposed framework automatically learns embedding change probabilities for every pixel in a given spatial cover image. Then, a generator G produces stego images according to the minimal embedding distortion calculated based on the change probabilities while a discriminator D distinguishes the produced stego images from the cover images. However, the performance of this type of method is no better than that of some state-of-the-art hand-crafted steganography algorithms.

The steganography methods with GANs in the methods described above are embedding based. In this paper, we propose an SWE method that applies DCGANs.

III. The Proposed Image Steganography Without Embedding

As illustrated in Fig. 1, the proposed steganography framework consists of three phases. In first phase, we train DCGANs on an image set and obtain generator G after DCGANs convergence. The network parameters of G are determined after the first phase, and the cover images are produced by G. During the second phase, we train a CNNs model, called the extractor E, based on the recovery errors from a large number of random noise vectors. We use G to extract information from stego images produced by G. In the third phase, the sender and the receiver hold the network and parameters of G and E, respectively. The sender divides the secret information into segments S_i, maps the segments into vectors z_i, and generates stego images by G according to z. After the receiver receives the stego images, who uses E to extract vector z_i' and then restores the secret data from z_i'. The main notation used in this section is given in Table II.

A. Cover Image Generation

The phase of cover image generation includes two main steps. In the first step, we divide the secret information S into segments S_i and then map each segment S_i to noise vector z_i. In the second step, we generate a cover image $stego_i$ (which

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

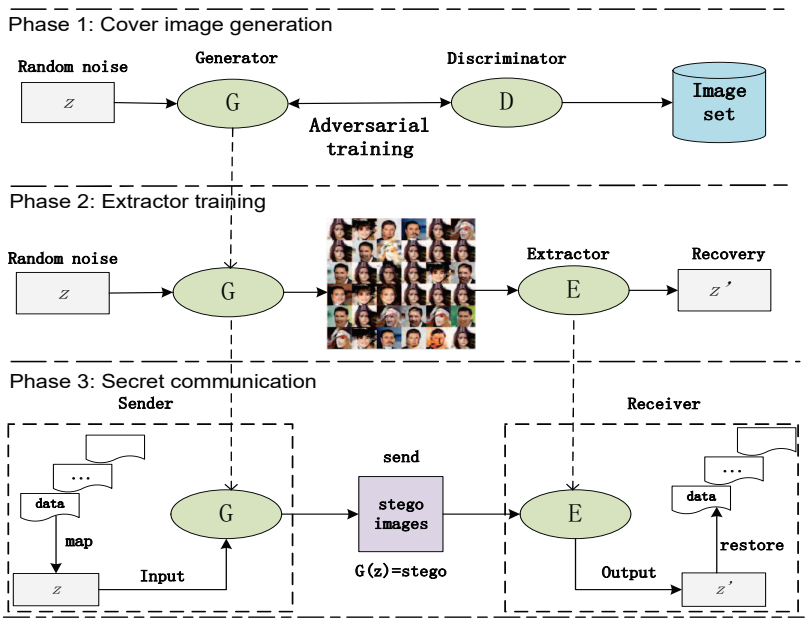


Fig. 1. The proposed steganography framework using DCGANs for SWE.

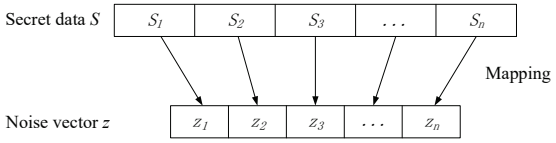


Fig. 2. Diagram of mapping secret data into vectors.

is also a stego image because the scheme is an SWE method) from the noise vector z_i with the help of DCGANs. In the mapping procedure, several bits (2 or three) of the segment are mapped to a noise value with a given interval according to the following equation:

$$r = random(\frac{m}{2^{\sigma-1}} - 1 + \delta, \frac{m+1}{2^{\sigma-1}} - 1 - \delta), \qquad (1)$$

where the function random(x,y) denotes a random noise value produced between x and y, r is the mapped noise vector within the interval $(\frac{m}{2^{\sigma-1}} - 1 + \delta, \frac{m+1}{2^{\sigma-1}} - 1 - \delta)$, m is the value of the secret data bits to be mapped, and σ is a positive integer variable that represents the number of secret data bits carried by one bit of random noise, $\sigma = |S_i|/|z_i|$. δ is the gap between the divided intervals, which allows a deviation tolerance when extracting data from a stego image and ensures the extraction accuracy of the secret data during the secret communication phase.

To further illustrate how the formula works, we present the details in Table I when σ is 3 and δ is 0.001. We convert every three bits of payload into random noise within an interval between -1 and 1. Eight intervals are obtained from (-1, 1) with a gap of 0.001, and each payload value from '000' to '111' corresponds to an interval. Therefore, the steganography capacity of each stego is $\sigma \times |z_i| = 3 \times |z_i|$.

The size of stego images generated from G is limited by the scale of the CNNs in DCGANs: the larger the dimension of noise is, the richer the details the generated image contains [21]. Secret information is converted into corresponding noise vectors and fed into the generator in the DCGANs to produce stego images.

TABLE I
RELATIONSHIP BETWEEN NOISE AND SECRET INFORMATION WITH 300 BIT STEGANOGRAPHY CAPACITY

payload	noise value
000	random(-0.999,-0.751)
001	random(-0.749,-0.501)
010	random(-0.499,-0.251)
011	random(-0.249,-0.001)
100	random(0.001,0.249)
101	random(0.251,0.499)
110	random(0.501,0.749)
111	random(0.751,0.999)

The structures of the generator G and discriminator D in DCGANs are introduced in [21]. G and D both are CNNs. G consists of a fully connected layer and four deconvolution layers and is used to fit the data distribution of the real images in the training set to produce an artificial image. D consists of four convolution layers and a fully connected layer. The input of D is an image with dimensions of 64×64×3, followed by four convolutional layers where the image dimensions are halved, and the number of channels is doubled from the previous layer. The last layer is a two-class softmax that outputs the probability of the image is from G or the training image set. The two CNNs are trained according to the loss calculated based on the feedback of each other. We train DCGANs to optimize the objective function defined as Eq.(2) [21], where D and G represent the generator and discriminator in the DCGANs, respectively. z is the noise vector from which G produces an artificial image, and x denotes images from the

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

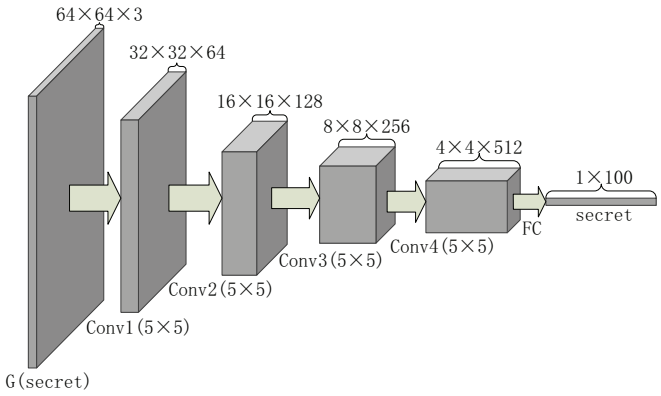


Fig. 3. The structure of extractor E.

image set.

$$\min_G \max_D V(G,D) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_z(z)}[log(1-D(G(z)))] \quad (2)$$

$$stego = cover = G(z) \quad (3)$$

We train DCGANs on an image set to obtain the generator model from which the stego image is derived. We map the preprocessed secret information into noise vectors, which are used as the input of G to generate stego images. Then G is used to convert the secret information into an image carried by the payload. In other words, there is a corresponding relationship between stego images and the secret information. Eq.(3) indicates that stego images are driven directly by secret data with the help of G, and they are not modified to hide information. Thus, the stego images and cover images are the same.

B. Training of the Extractor

We design the CNNs, called the extractor E, to recover the secret data from stego images generated by G. We study the architecture of D to accurately recover the information from E. Similar to the structure of D, E has four convolutional layers and a fully connected layer, as shown in Fig. 3.

The input of E is a stego image of dimensions 64×64×3, and the output is a noise vector with dimensions of 1×100. We change the function in the output layer of D from softmax to tanh. The output of the tanh activation function satisfies the condition that the noise values must be between -1 and 1. In CNNs, the dropout operation, activation function and pooling layer are used to enhance the nonlinear learning capabilities of neural network. The purpose of the CNNs are to use nonlinear features to learn the fitting parameters.The weight parameters in each layer of the network are learned to fit the mapping between the input and output. The effect of CNNs is similar to that of linear multivariate equations if the roles of these nonlinear operations are ignored. From this perspective, we design E with a network structure that is opposite to that of the generator. We use a leak-Relu activation function and batch normalization in each layer with no pooling layer or dropout operation. Additionally, a fully connected layer is used after the last convolutional layer. We train E to extract information from the generated stego images from G. The training procedure of the extractor is illustrated as phase 2 in Fig. 1.

$$L(E) = \sum_{i=1}^{n}(z - E(stego))^2 = \sum_{i=1}^{n}(z - E(G(z))^2 \quad (4)$$

We constantly generate random noise vectors between -1 and 1 with the same dimensions as z_i as the input of G. In addition, to avoid over-fitting when E extracts information from stego images generated by certain noise vectors, the noise vectors fed to extractor E are generated with different random seeds. Then, the generator produces cover images as the input of the extractor according to these noise vectors. E outputs z' as the recovery of z. The optimization goal for training E is to minimize the deviation between noise z and recovery z' resolved by the extractor. The loss function of the extractor model is defined as the square loss, which is shown in Eq.(4), where z represents the noise vector and E denotes the extractor. E($stego$) is the noise vector recovered from the stego image by the extractor, and G(z) is the image generated by G from noise z. We train E for a specific G obtained from different image databases. When the loss of training is sufficiently small or the network of E is convergent, we use E to recover secret data from the stego images produced by G.

C. Secret Communication

As illustrated in phase 3 of Fig. 1, the conditions for secret communication by the proposed method are that the sender holds the CNNs model G and the corresponding network parameters of G and the receiver holds the CNNs model E and the corresponding network parameters of E. The detailed hiding algorithm is shown in Algorithm 1. After training the DCGANs, the sender divides the secret information into segments with the length of the steganography capacity of the stego image. The sender calculates the value m of every σ-bits binary numbers in each information segment and then maps the σ-bits into a value in the noise vector according to the mapping rule defined in Eq.(1). The sender sends these noise vectors to the input of generator G to produce the corresponding stego images. The sender transmits all the stego images in turn. The recovery algorithm is shown in Algorithm 2. The receiver receives the stego images, uses E to extract the noise vectors, and restores the secret data according to the reverse mapping rules.

IV. Experimental Results and Analysis

We trained the DCGAN model on two datasets[1]: Celebrities, which contains 200k face images, and Food101, which contains 50k food images. All the images in the two datasets are cropped to 64×64, and the models are trained with mini-batch stochastic gradient descent (SGD) with a mini-batch size of 100. The generated images are 64×64.

[1]Celebrities is available at http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html. Food101 is available at http://www.vision.ee.ethz.ch/datasets_extra/food-101/.

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

TABLE II
NOTATION USED IN THIS PAPER

Symbol	Description	Symbol	Description
S	Secret information	S_i	The ith segment of S
S_{ij}	The jth bit of S_i	l	Length of the secret data in each stego image
$random(x, y)$	Function generating random numbers between x and y	t	Length of noise vector z
σ	Coefficient of mapping between noise and secret data (positive integer)	δ	Gap between the intervals
z	Noise vectors	z_i	The ith noise vector in z
z_{ij}	The jth value of z_i	n	The number of stego images
$G(z)$	Generator G	$E(m)$	Extractor E
$stego$	Total stego images	$stego_i$	The ith stego image
m	Value of secret information bits mapped by each bit of noise	r	One bit of noise vector mapped by secret data

Algorithm 1 Cover (stego) Image Generation

Input: Variables:S, t, σ, δ.
Output: $stego$

train the DCGANs on an image set to obtain generator G by using Eq.(2);
$l = \sigma \times t$;
$n = \lceil length(S)/l \rceil$;
divide secret information into n segments with length of l;
for $i = 1$ to n **do**
  //loop will iterate for all secret information segments from 1 to n;
  for $k = 1$ to l **do**
    //loop will iterate for all bits of each secret information segment;
    $m = 0$;
    for $j = k$ to $k + \sigma - 1$ **do**
      $m = m + 2^{k+\sigma-1-j} S_{ij}$;
    end for
    $k = k + \sigma$;
    generate r by using Eq.(1);
    insert r into z_i;
  end for
  insert z_i into z;
end for
for $i = 1$ to n **do**
  input z_i into DCGANs and get $stego_i = G(z_i)$;
  insert $stego_i$ into $stego$;
end for
return $stego$.

Algorithm 2 Secret Data Extraction and Recovery

Input: Variables:$stego, t, \sigma$.
Output: S

train the extractor E according to the generator G by using Eq.(4);
$n = length(stego)$;
for $i = 1$ to n **do**
  //loop will iterate for all stego images
  $z_i = E(stego_i)$;
  $m = 0$
  for $j = 1$ to t **do**
    $m = \lfloor (z_{ij} + 1) \times 2^{\sigma-1} \rfloor$;
    //recover secret data according to the reverse mapping rules
    insert binary bits with value of m into S_i;
  end for
  insert S_i into S;
end for
return S.

We hope that the learning rate is generally set according to the number of training rounds. From the change in loss in the experiments, we found that it is suitable to set the initial learning rate to 0.0002. The learning rate after a certain number of training rounds should be gradually reduced; thus, we use Adam optimization [36] to train the DCGANs and the extractor E, and we use the steganography capacity and the resistance to detection by steganalysis and forensics methods to evaluate the SWE methods. Because we generate stego images by using DCGANs and recover the information from stego images by using CNNs, we analyze the anti-forensic to

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

image tempering of stego images and the accuracy of secret data recovery from stego images. The experiments consist of five parts. In the first part, we train the DCGANs on the two image sets to obtain the generator to create cover images. In the second part, we train the extractor E based on the output of the convergent generator G. In the third part, we test the secret information recovery accuracy by using the convergent extractor. In the fourth part, we compare the steganography capacity of our method with that of several state-of-the-art SWE methods. Finally, we analyze the security of the proposed SWE method.

In our method, we used TensorFlow running in GPU mode while computing gradient. In the training phase, the model is run on a machine with an NVIDIA gtx1070 GPU and 128GB of memory. We trained the DCGANs for 300 epoches on two datasets, Celebrities and Food101, for 6 days and 4 days, respectively.

A. Training of DCGANs

The training of DCGANs is carried out on the Celebrities and Food101 image training sets. The dimension of the noise vector z, the input of the generator, is 100. In this section, we focus on how to obtain a generator G that can produce high-quality cover images. We optimize DCGANs based on the objective function defined in Eq.(2). In the experiments, we set the learning rate to 0.0002 and the mini-batch to 100. For each image sets, DCGANs are trained in 300 epochs. Some details of a mini-batch sample output from G are shown in Fig. 4. Fig. 4(a) shows samples trained on Celebrities, and Fig. 4(b) shows samples trained on Food101. After training, due to the characteristic of DCGANs, the output image fits the data distribution of the images in the corresponding training set. Fig. 4(a) and Fig. 4(b) show artificial face and food images produced by G, respectively.

B. Training of the Extractor

In this section, we train the extractor with a large number of random noise vectors. The noise values are uniformly distributed in (-1, 1). We optimize the extractor by training as much as possible to improve its accuracy. The differences between the outputs from E and the original noise vectors represent the recovery accuracy; thus, we take the square loss as the training loss function. In the training procedure, the inputs to the extractor are images of size 64×64, the learning rate is set to 0.0002, and the size of a mini-batch is set to 100. In each training mini-batch, a 100×100 random noise matrix is produced as the input of the generator, from which mini-batch images are generated. Then, these images are sent to the input of the extractor as stego images. The loss is calculated from the noise matrix and the outputs from extractor via the loss function, and Adam optimization is used in training. We train extractor E based on the convergent generator G trained separately on the two training sets. We record the loss value after every 100 batches, which is called a step. The loss curves with steps of training are shown in Fig. 5 and Fig. 6.

For conciseness, we call the curve shown in Fig. 5 loss a and the curve shown in Fig. 6 loss b. Loss a and loss b show

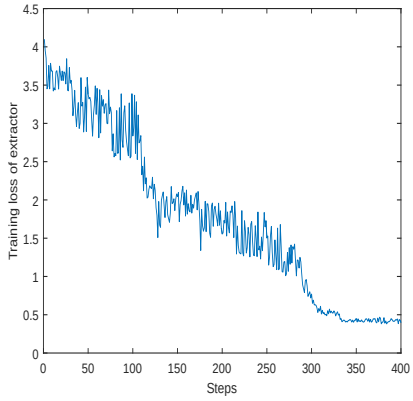


Fig. 5. The training loss of extractor based on generator G trained on Celebrities.

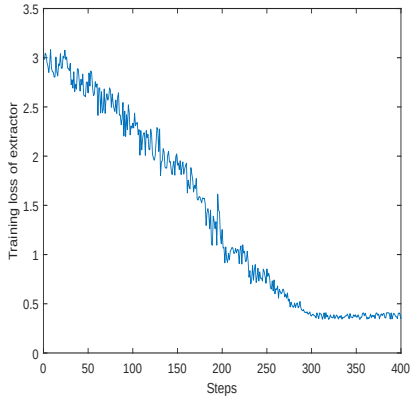


Fig. 6. The training loss of extractor based on generator G trained on Food101.

that the loss value of E for generator G trained on Celebrities is approximately 3.7 while the loss value of E for generator G trained on Food101 is approximately 3 at the beginning of training. Loss a and loss b fluctuate violently initially, which means that extractor's ability to resolve secret information is weak. However, the losses gradually decrease as more training steps are completed, indicating that the training of extractors to resolve information based on different generators G is effective. Although loss a fluctuates in a larger range than that of loss b, their convergence speeds are equivalent. After approximately 300 steps, loss a and loss b both become stable at approximately 0.3. Therefore, the losses of extractor E based on the two training sets reach the same value when E converges, despite the differences in training procedures. Thus, the extraction ability of extractor E is not dependent on the image set but on the CNNs structure of generator G. After 300 steps, there is no significant decline in loss a and loss b.

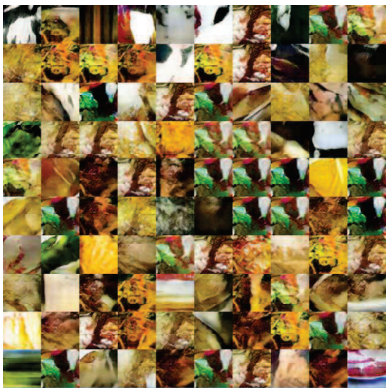
C. Accuracy of Secret Data Recovering

The extraction accuracy of the noise vector is the key to recovering secret data. To assess the accuracy of data recovery under different conditions, we conduct experiments to recover secret data from the output of extractor E under different σ and δ values. We use G obtained by training DCGANs to generate

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

(a) Samples generated by G trained on Celebrities.



(b) Samples generated by G trained on Food101.

Fig. 4. Mini-batch generated images output from G trained on different images.

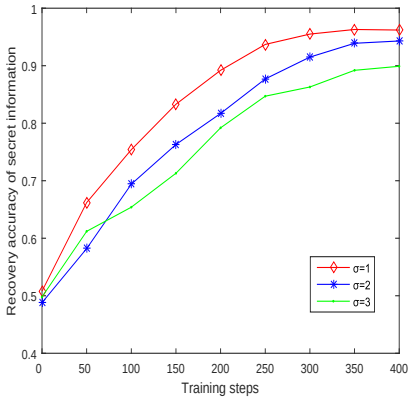


Fig. 7. Recovery accuracy of secret information extracted from stego images by using E trained on the Celebrities image set.

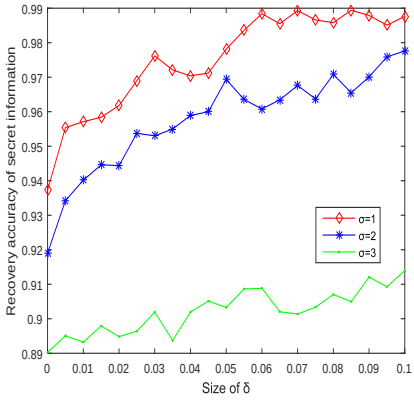


Fig. 9. Recovery accuracy of secret information extracted from stego images under different δ on the Celebrities image set.

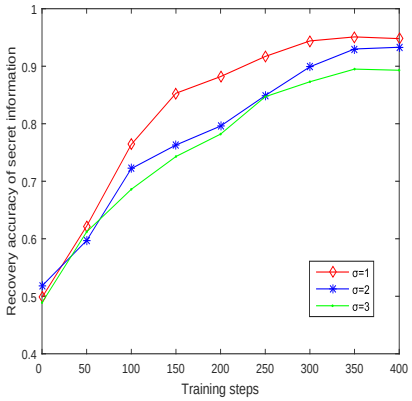


Fig. 8. Recovery accuracy of secret information extracted from stego images by using E trained on the Food101 image set.

stego images and use E to recover secret data according to the reverse mapping rule.

To illustrate the impact of the number of training steps on the accuracy of the extractor, we set δ to 0.01 and vary σ from 1 to 3 and conduct extraction experiments with the E obtained by training. The recovery accuracy is defined as the ratio of the number of binary bits recovered correctly from the stego image to the length of the original secret information. Fig. 7 and Fig. 8 show the curves of the recovery accuracy under different σ values. The recovery accuracy increases as the number of training steps increases. After 300 steps, the recovery accuracy exceeds 0.90 when σ is 1 and 2 and reaches approximately 0.96 when σ is 1. When σ is 3, the recovery accuracies for the two training sets are 0.893 and 0.887. Additionally, the recovery accuracy decreases with increasing σ. In other words, the smaller the payload is, the higher the extraction accuracy becomes.

To demonstrate the impact of δ on the recovery accuracy, we vary σ from 1 to 3 and use E after 300 epochs to recover secret data under different δ. The recovery results are shown in Fig. 9 and Fig. 10. σ is the index representing the steganography capacity. The results show that the recovery accuracy increases with increasing δ under different steganography capacities. When σ is 1 and δ is 0.1, the recovery accuracy is approximately 0.98. We also perform experiments to illustrate the impact of σ on the recovery accuracy; the results are shown in Fig. 11, where δ is 0.01 and σ is varied from 1 to 5. As the steganography capacity increases, the recovery accuracy decreases. By comparison, the recovery accuracy of method

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

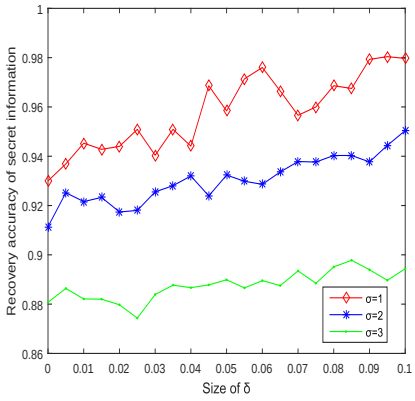


Fig. 10. Recovery accuracy of secret information extracted from stego images under different δ on the Food101 image set.

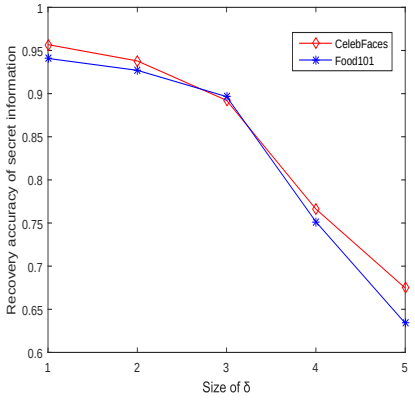


Fig. 11. Recovery accuracy from stego images under different steganography capacities.

in [25] is less than 0.87 when the payload is 0.4 bpp, which is lower than our recovery accuracy.

D. Steganography Capacity

Currently, the steganography capacities of SWE methods are much lower than those of traditional embedding-based steganography. Because our method is a new SWE method, in this paper we simply compare the steganography capacity with other state-of-the-art SWE methods, including the cover-selection-based and the cover-synthesis-based methods. The comparison results are shown in Table III, where the second column is the absolute steganography capacity (steganography capacity per image), the third column is the the size of the stego image, and the last column is the relative steganography capacity (steganography capacity per pixel):

$$Relative\ capacity = \frac{Absolute\ capacity}{The\ size\ of\ the\ image} \tag{5}$$

The relative capacity of our method is 9.16e-3 bytes/pixel, which is shown in the last row of Table III. Rows 1-3 show the capacities of cover-selection-based methods. Apparently, the relative capacity of our methods is much greater than those of cover-selection-based methods. Rows 4-6 show the capacities

TABLE III
STEGANOGRAPHY CAPACITIES OF SWE METHODS

Methods	Absolute capacity (bytes/image)	Image size	Relative capacity (bytes/pixel)
method in [13]	1.125	512×512	4.77e-6
method in [12]	3.72	⩾512×512	1.42e-5
method in [14]	2.25	512×512	8.58e-6
method in [35]	25∼100	480×640	8.14e-5∼3.26e-4
method in [15]	64×64	800×800	6.40e-3
method in [16]	1535∼4300	1024×1024	1.46e-3∼4.10e-3
ours	⩾37.5	64×64	9.16e-3

of cover-synthesis-based methods. Although compared with the relative capacities of the cover-selection-based methods the relative capacities of cover-synthesis-based methods are improved significantly, they are still lower than our capacities. In other words, our method outperforms state-of-the-art SWE methods, including the cover-selection-based and cover-synthesis-based methods.

E. Security Analysis

In the proposed method, there is no way to reveal the information concealed in the stego images except with the extractor E. Hence, we consider two aspects of security: the resistance to detection of steganalysis and the anti-forensics of generated stego images. Due to the characteristics of SWE, there is no difference between cover images and stego images. Thus, SWE can effectively resist the detection of steganalysis algorithms. By contrast, the stego images are generated by CNNs in our method. To prove that the stego images in our method can also resist detection by steganalysis algorithms, experiments are conducted on the detection of stego images produced by our method via state-of-the-art image steganalysis algorithms. Since the stego images generated by CNNs are computer-generated images, which are unnatural images even though they look like natural images, the ability of the proposed method to resist detection by image forensics algorithms is also considered in this section.

We generate 5000 stego images via the proposed method as samples and then detect them with different steganalysis and image forensic algorithms. The probability of being identified as a stego image or tampered image is shown in Table IV. The CRM proposed in [7] is an extension of the spatial rich model for steganalysis of color images. Ni's model [37] is a CNN-based steganalysis algorithm for steganographic algorithms in the spatial domain. Median filtering is among the most common manipulation methods in image processing. The image forensics algorithm that we use, CMFF, which is a CNN-based forensics algorithm with image median filtering, was proposed in [38].

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

TABLE IV
THE PROBABILITY OF IDENTIFYING A STEGO IMAGE OR TAMPERED IMAGE UNDER DETECTION BY DIFFERENT ALGORITHMS

Detection algorithms	CRM [7]	Ni's model [37]	CMFF [38]
First case	0.008	0.47	0
Second case	0.56	0.98	0.74

In the experiment, we consider two different cases. In the first case, the training data set of the steganalysis methods [7] [37] consists of the cover images used in our method and the stego images generated using embedding-based algorithms, and the training data set of [38] consists of the cover images used in our method and the splicing image generated by adding media filtering. Thus, the training image sets of [7] [37] [38] are different from our image sets, and their detection accuracies are shown in the first row of Table IV, from which we can see that their detection accuracies are low (0.008, 0.47 and 0, respectively). In the second case, the training sets of [7] [37] [38] are same as ours (that is to say, in [7] [37] we use a natural image in our model as the cover images, and we use the generated images in our model as the "embedded stego" images; in [38] we use the natural image in our model as the natural images, and we use the generated images in our model as the tampered images). Their detection accuracies are shown in the second row of Table IV, from which we can see that the detection accuracies of the three algorithms are not low (0.56, 0.98 and 0.74, respectively).This means that when directly using our training set to train the classifies in [7] [37] [38], they achieve good detection ability. However, in real-world applications, we can keep the training set of our method as a secret, thus ensuring security in terms of resisting detection by steganalysis and forensics methods.

F. Further Discussion

To generate more realistic stego images with high steganography capacity, we can change the structure of DCGANs, such as increasing the length of the noise vectors, and changing the depth and parameters of the CNNs. In this paper, we cannot ensure that secret information is recovered with one hundred percent accuracy; the same question exists in [25]. The reason for this problem is that instead of recovering secret information according to human-designed extraction rules (which are used in the traditional embedding-based steganography schemes), we train the CNNs to recover secret information. The loss of the CNNs is negligible when used for classification, but there is a loss in recovery accuracy when the CNNs are used for steganography extraction. We may further improve the extraction accuracy with the help of Reed-Solomon error-correction codes. We can calculate the check codes of all secret message segments and hide these check codes into additional stego images. The parameters (σ and δ) can be changed to ensure that the check codes contained in these stego images are recovered correctly (however, it may cause the additional images to look similar).

V. CONCLUSIONS

This paper proposes a new image steganography method based on stego images generated by DCGANs according to secret information.In other words, we build a functional relationship between secret information and stego images without embedding by using CNNs. Moreover, CNNs model that can successfully extract secret information from stego images is proposed. The imperceptibility of secret information by this method is significantly improved such that the image steganography can effectively resist detection by steganalysis and forensics algorithms. We apply DCGANs to image steganography, so the disadvantages of DCGANs result in some drawbacks. For example, some generated stego images are not sufficiently natural to completely escape detection by steganalysis, the size of the stego image is small, and the steganography capacity is not big adequate. These problems can be resolved with the development of more powerful neural networks. The recovery accuracy of our method is not perfect, but this problem can be resolved by including error-correction codes in our method. These problems will be addressed in future work.

REFERENCES

[1] C. Cachin, *An Information-Theoretic Model for Steganography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 306–318. [Online]. Available: https://doi.org/10.1007/3-540-49380-8_21

[2] T. Pevn, T. Filler, and P. Bas, "Using high-dimensional image models to perform highly undetectable steganography," *Lecture Notes in Computer Science*, vol. 6387, pp. 161–177, 2010.

[3] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *Eurasip Journal on Information Security*, vol. 2014, no. 1, p. 1, 2014.

[4] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *IEEE International Workshop on Information Forensics and Security*, 2013, pp. 234–239.

[5] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics & Security*, vol. 7, no. 3, pp. 868–882, 2012.

[6] J. Fridrich, "Quantitative steganalysis using rich models," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 8665, pp. 86 650O–86 650O–11, 2013.

[7] M. Goljan, J. Fridrich, and R. Cogranne, "Rich model for steganalysis of color images," in *IEEE International Workshop on Information Forensics and Security*, 2015, pp. 185–190.

[8] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont, "Deep learning for steganalysis is better than a rich model with an ensemble classifier, and is natively robust to the cover source-mismatch," *CoRR*, vol. abs/1511.04855, 2015. [Online]. Available: http://arxiv.org/abs/1511.04855

[9] J. Zeng, S. Tan, B. Li, and J. Huang, "Large-scale

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2018.2852771, IEEE Access

jpeg steganalysis using hybrid deep-learning framework," *arXiv preprint arXiv:1511.04855*, 2016.

[10] Y. Qian, J. Dong, and W. Wang, "Deep learning for steganalysis via convolutional neural networks," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 9409, pp. 94 090J–94 090J–10, 2015.

[11] M. Barni, "Steganography in digital media: Principles, algorithms, and applications [book reviews]," *IEEE Signal Processing Magazine*, vol. 28, no. 5, pp. 142–144, 2011.

[12] Z. L. Zhou, Y. Cao, and X. M. Sun, "Coverless information hiding based on bag-of-words model of image," *Journal of Applied Sciences*, vol. 34, no. 5, pp. 527–536, 2016.

[13] Z. Zhou, H. Sun, R. Harit, X. Chen, and X. Sun, "Coverless image steganography without embedding," in *International Conference on Cloud Computing and Security*, 2015, pp. 123–132.

[14] S. Zheng, L. Wang, B. Ling, and D. Hu, *Coverless Information Hiding Based on Robust Image Hashing*. Cham: Springer International Publishing, 2017, pp. 536–547. [Online]. Available: https://doi.org/10.1007/978-3-319-63315-2_47

[15] J. Xu, X. Mao, A. Jaffer, A. Jaffer, S. Lu, L. Li, and M. Toyoura, "Hidden message in a deformation-based texture," *Visual Computer International Journal of Computer Graphics*, vol. 31, no. 12, pp. 1653–1669, 2015.

[16] K. C. Wu and C. M. Wang, "Steganography using reversible texture synthesis," *IEEE Transactions on Image Processing*, vol. 24, no. 1, p. 130, 2015.

[17] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *International Conference on Neural Information Processing Systems*, 2014, pp. 2672–2680.

[18] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *Computer Science*, pp. 2672–2680, 2014.

[19] E. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 1486–1494. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969239.2969405

[20] M. L. K. M. G. Bousquet, "Are gans created equal a large-scale study," *arXiv:1711.10337v2*, 2017.

[21] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *Computer Science*, 2015.

[22] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, pp. 1060–1069, 2016.

[23] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic, "Generating images with recurrent adversarial networks," *arXiv preprint arXiv:1602.05110*, 2016.

[24] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with perceptual and contextual losses," *arXiv preprint arXiv:1607.07539*, 2016.

[25] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," *arXiv preprint arXiv:1703.00371*, 2017.

[26] D. Volkhonskiy, B. Borisenko, and E. Burnaev, "Generative adversarial networks for image steganography," *ICLR 2016 Open Review*, 2016.

[27] W. Tang, S. Tan, B. Li, and J. Huang, "Automatic steganographic distortion learning using a generative adversarial network," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–1, 2017.

[28] J. Mielikainen, "Lsb matching revisited," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 285–287, 2006.

[29] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *Information Forensics & Security IEEE Transactions on*, vol. 5, no. 2, pp. 215–224, 2010.

[30] J. Fridrich, M. Goljan, and R. Du, "Detecting lsb steganography in color, and gray-scale images," *Multimedia IEEE*, vol. 8, no. 4, pp. 22–28, 2001.

[31] Y. Q. Shi, C. Chen, and W. Chen, "A markov process based approach to effective attacking jpeg steganography," in *Information Hiding, International Workshop, Ih 2006, Alexandria, Va, Usa, July 10-12, 2006. Revised Selcted Papers*, 2007, pp. 249–264.

[32] T. Pevny and J. Fridrich, "Merging markov and dct features for multi-class jpeg steganalysis," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 6505, pp. 650 503–650 503–13, 2007.

[33] D. Hu, Q. Shen, S. Zhou, X. Liu, Y. Fan, and L. Wang, "Adaptive steganalysis based on selection region and combined convolutional neural networks," *Security and Communication Networks*, vol. 2017, no. 4, pp. 1–9, 2017.

[34] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009, pp. 1–20.

[35] H. Otori and S. Kuriyama, "Texture synthesis for mobile data communications," *IEEE Computer Graphics and Applications*, vol. 29, no. 6, pp. 74–81, 2009.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.

[37] J. Ye, J. Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2545–2557, 2017.

[38] J. Chen, X. Kang, Y. Liu, and Z. J. Wang, "Median filtering forensics based on convolutional neural networks," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1849–1853, 2015.

2169-3536 (c) 2018 IEEE. Translations and content mining are permitted for academic research only. Personal use is also permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.